**Application Fraud Analysis**

University of Southern California

DSO 562 Fraud Analytics

Spring 2020

Prepared By:

Aixuan Liu [aixuanli@usc.edu](mailto:aixuanli@usc.edu)

Chao Wang [wang724@usc.edu](mailto:wang724@usc.edu)

Cheng Shi [shi005@usc.edu](mailto:shi005@usc.edu)

Chengjun Liu [cliu4428@usc.edu](mailto:cliu4428@usc.edu)

Minglu Chi [mingluch@usc.edu](mailto:mingluch@usc.edu)

Shijie Xiang [shijiex@usc.edu](mailto:shijiex@usc.edu)

**Table of Contents**

#### 

[**Executive Summary**](#_tiyzyv5qxcmx) **3**

[**Description of Data**](#_71vspvx9i7ry) **4**

[Field Summary](#_fhs0zniznlhp) 4

[Field Descriptions](#_6o5bem68nxqm) 5

[**Data Cleaning**](#_mf162vkw8flp) **9**

[**Creating Candidate Variables**](#_e5j30wne1uge) **11**

[Combining Related Fields](#_g0unig8owru3) 11

[Creating Candidate Variables](#_hk8ue7qh8t8x) 12

[Data Normalization](#_5mswbww8qzv3) 14

[Splitting Data](#_fa9smar1q1hx) 15

[**Feature Selection**](#_lhet0pqdy1no) **16**

[Univariate Filters](#_bg355inl00yp) 16

[Wrapper](#_ssc70fr0x82w) 17

[**Model Algorithms**](#_pi9sk18h0xf9) **18**

[Logistic Regression](#_2xqlh91laemm) 18

[Boosted Tree](#_7w564iis33g9) 18

[Random Forest](#_5mfik04v6zxq) 19

[Neural Net](#_dwgxrfccha3n) 20

[**Results**](#_49affc4cp8cv) **23**

[**Conclusion**](#_k9n07s3hkpzw) **26**

[Appendix](#_7x6p9vcpmnz0) **27**

#### 

#### Executive Summary

The purpose of this project is identifying fraud applications in the application data using supervised machine learning algorithms. The data is a statistical replication of real application data, with personally identifiable information removed and statistical qualities preserved. The data contains 1 million records from 2016-01-01 to 2016-12-31, with 10 categorical fields including a fraud label column.

The report begins with a data quality assessment, from which we see that a number of fields contain frivolous values of high occurrence due to sloppy data entry from applicants. This problem is addressed in data cleaning so that frivolousness would not be confused as fraud during model building.

The next section describes the process of candidate variable creation. The goal of variable creation is to capture the characteristics of common application fraud in terms of volume and velocity. Three kinds of variables, 286 in total, are created. Then all candidate variables are normalized by Z-scaling.

Next the report details the process of feature selection. A two-step approach is employed. First, two univariable filters, a KS score and Fraud Detection Rate (FDR) at 3%, are used to rank all candidate variables. Then, a backward selection wrapper algorithm, Recursive Feature Elimination and Cross-Validation Selection, is run several times on the top 143 candidate variables to reduce the number to 20. After feature selection, the original data file with only the selected features is splitted into training, testing, and out-of-time sets for model building and evaluation.

To detect fraud, a linear logistic regression is trained on the training set as the base model, and 3 nonlinear classification models, namely Boosted Tree, Random Forest, and Neural Net, are built with a range of parameters. The best performing model on the testing set, Random Forest, is selected as the final model, and three FDR tables are built with training, testing, and out-of-time sets separately for evaluation.

#### 

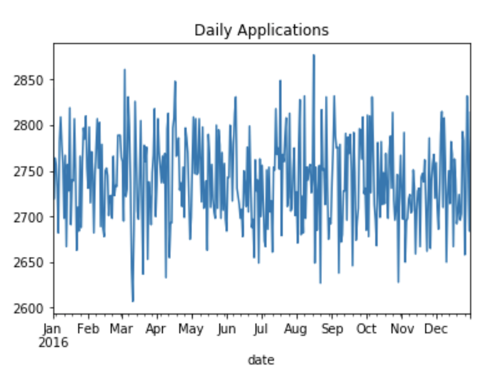
#### Description of Data

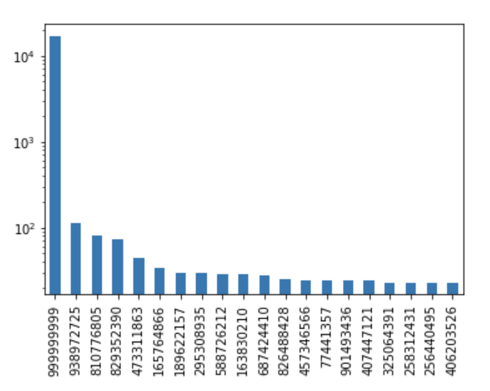
The Application Fraud data is a dataset emulating the statistical nature of real fraud data. The data file contains 1 million records of applications in 2016, with a record number column, 8 columns of personal information, and one fraud label column indicating whether an application has turned out to be fraud.

##### Field Summary

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Field No. | Field Name | Data Type | Field  Type | # of Populated Records | % of Populated Records | # of Unique Values | # of Zeros | Most Common Field Value | Number of Occurrence |
| 1 | record | int64 | Categorical | 1M | 100% | 1M | 0 | N/A | N/A |
| 2 | date | int64 | Categorical | 1M | 100% | 365 | 0 | 2016-01-01 | 19,276 |
| 3 | ssn | int64 | Categorical | 1M | 100% | 835,819 | 0 | 999999999 | 16,935 |
| 4 | firstname | object | Categorical | 1M | 100% | 78,136 | 0 | EAMSTRMT | 12,658 |
| 5 | lastname | object | Categorical | 1M | 100% | 177,001 | 0 | ERJSAXA | 8,580 |
| 6 | address | object | Categorical | 1M | 100% | 828,774 | 0 | 123 MAIN ST | 1,079 |
| 7 | zip5 | int64 | Categorical | 1M | 100% | 26,370 | 0 | 68138 | 823 |
| 8 | dob | int64 | Categorical | 1M | 100% | 42,673 | 0 | 19070626 | 126,568 |
| 9 | homephone | int64 | Categorical | 1M | 100% | 28,244 | 0 | 9999999999 | 78,512 |
| 10 | fraud\_label | int64 | Categorical | 1M | 100% | 2 | 985,607 | 0 | 985,607 |

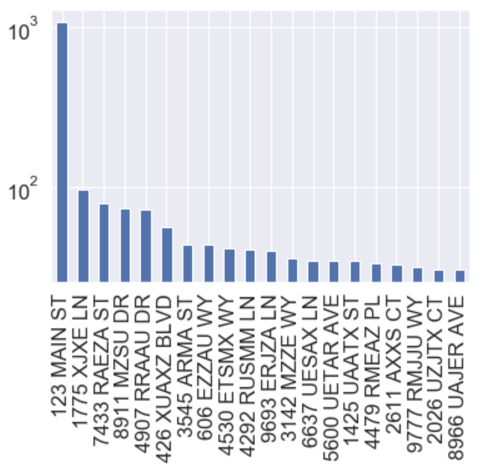
##### Field Descriptions

* + 1. date  
       Dates of the applications range from 2016-01-01 to 2016-12-31. Since the number of occurrences of each date is the same as the number of applications of each day, we plot the number of daily applications for an overview of the distribution of dates and applications. Since 2016 is not an ecliptic year, we fill the date 2016-02-29 with the same number of applications on 2016-02-28 to remove the dip in the data.
    2. ssn

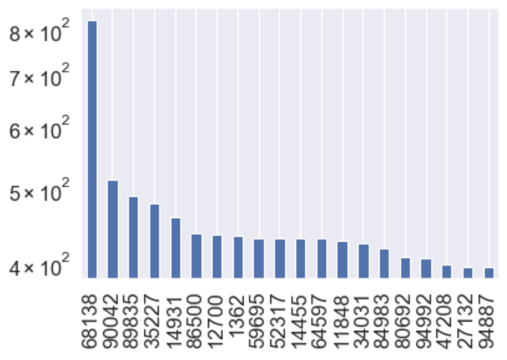
The social security number filled by applicants. The distribution of the top 15 SSNs with the most occurrences is as below. 999999999 has the most occurrences, which indicates that a lot of people filled in this field frivolously.

* + 1. address

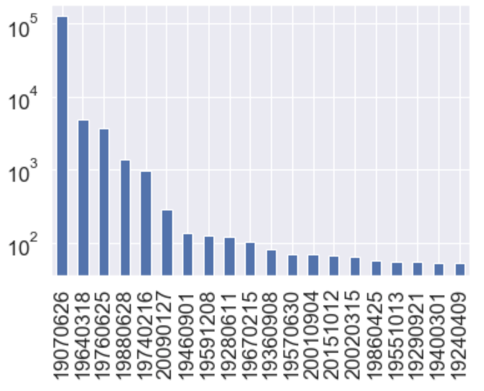
The addresses given by the applicants for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 123 MAIN ST is the most common address and has ten times more occurrences than the second ranking address. It is very likely to be a result of applicants filling in addresses frivolously.



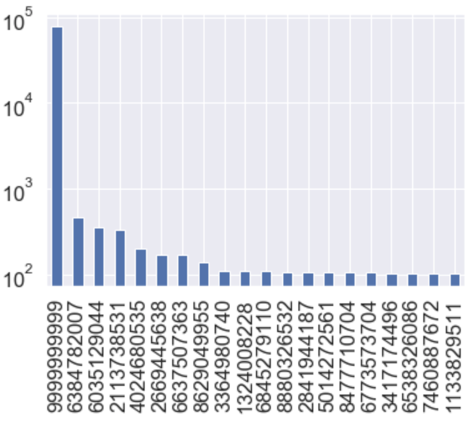
* + 1. zip5

The 5-digit zip code given by the applicant for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 68138 is the most common zip code and has significantly more occurrences than the second ranking zip code. It is likely to be a result of applicants filling in information frivolously, or an indication of possible fraud applications.

* + 1. dob

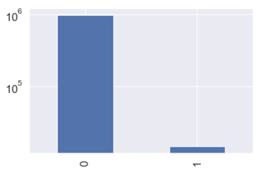
Date of birth given for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 19070626 is the most common date of birth. It is likely to be a result of applicants filling in information frivolously, or an indication of possible fraud applications.

* + 1. homephone

Landline at home given for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 9999999999 is the most common phone number. It is likely to be a result of applicants filling in information frivolously.

* + 1. fraud\_label

The indicator variable of whether the record later turned out to be a fraud application. Zeros indicate normal applications, and ones indicate fraud applications. The majority of the records were non-fraud applications, which accounts for 98.56% of the entire dataset. There are 14,393 fraud records in total, which account for about 1.44%.



#### Data Cleaning

After conducting data exploration, we have found that all fields are 100% populated, so there is no need to fill in missing values. As all fields are categorical, there are no extreme numeric or outlier values that need to be excluded.

From the data quality report, we can see that there are frivolous values occurring in variables “ssn”, “homephone” , “address” and “dob”. Frivolous data entry leads to a large number of occurrences of certain values, which can be problematic for fraud detection. Repetitive occurrence of fields and records is a strong indicator of fraud behavior, usually fraudsters attempting to game the system by trying out different combinations of personal information.

To make sure that frivolousness does not be counted as fraud behaviours, we replace the apparent frivolous values in a record with the value in its “record” field, which is the unique identifier of the record. Even though the replacement will change the format of the values, it does not affect fraud detection as real repetition is our major concern. Charts below show the distribution of the fields after cleaning:





#### 

#### Creating Candidate Variables

There are 10 fields in the dataset. Excluding “record”, “date” and “fraud\_label”, 7 fields can be used to generate candidate variables for model building. They fall into 2 categories: identity information (“firstname”, “lastname”, “ssn” and “dob”) and contact information (“address”, “zip5” and “homephone”).

We create candidate variables that can catch the most common modes of application fraud. They could either be fraudsters trying out a large number of illegally obtained personal information with a smaller number of contact information of the fraudsters themselves, or it could be one particular victim’s stolen personal information appearing in a number of applications with different fraudsters’ contact information. In addition, fraud applications usually happen in a burst. Therefore, we create three kinds of variables to catch the three characteristics of fraud:

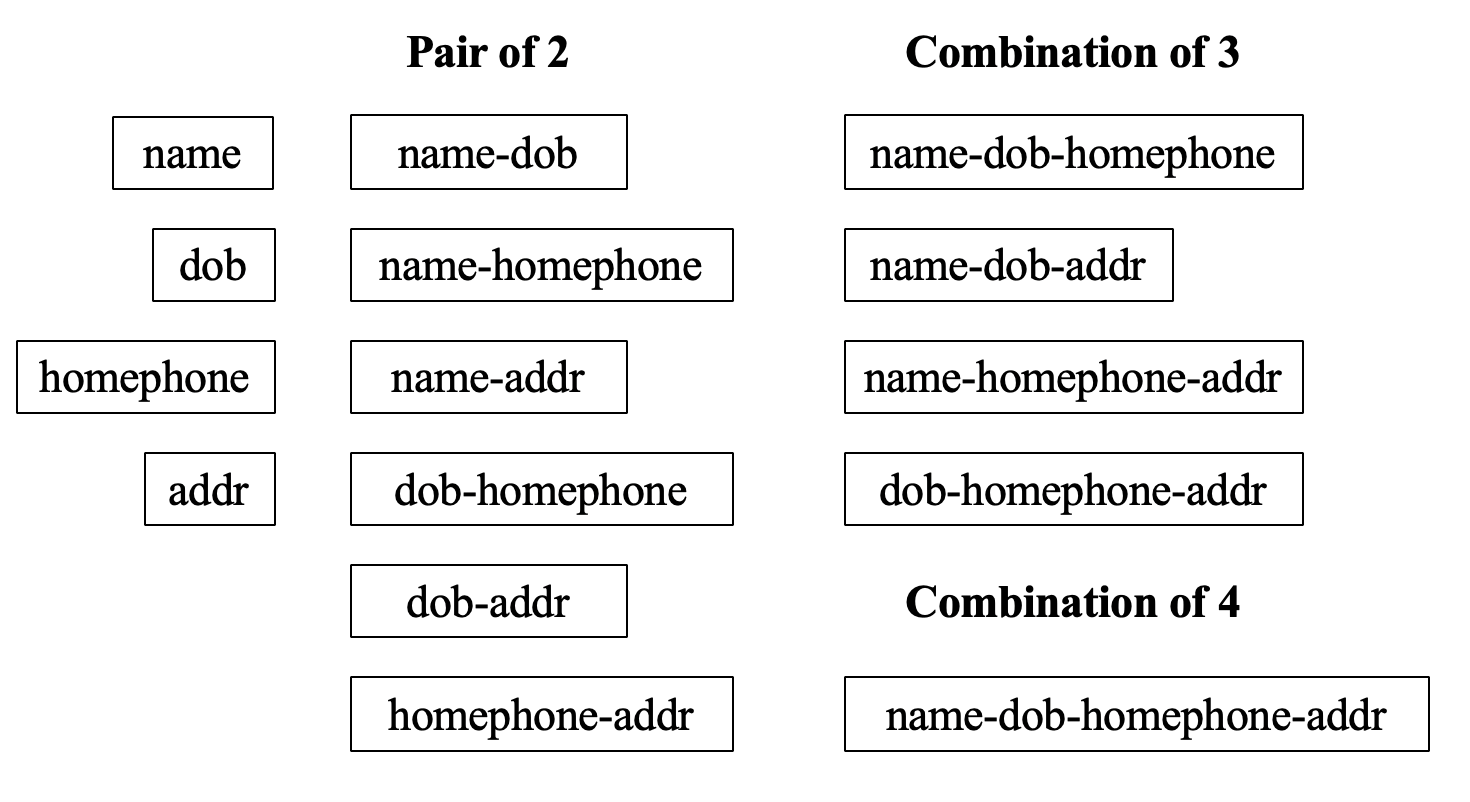
* Days Since: How many days since the the information was last seen
* Velocity: The number of applications with the same information within a specific time period
* Relative Velocity: The ratio of the number of applications with the same information in a short period of time versus a longer period of time

##### Combining Related Fields

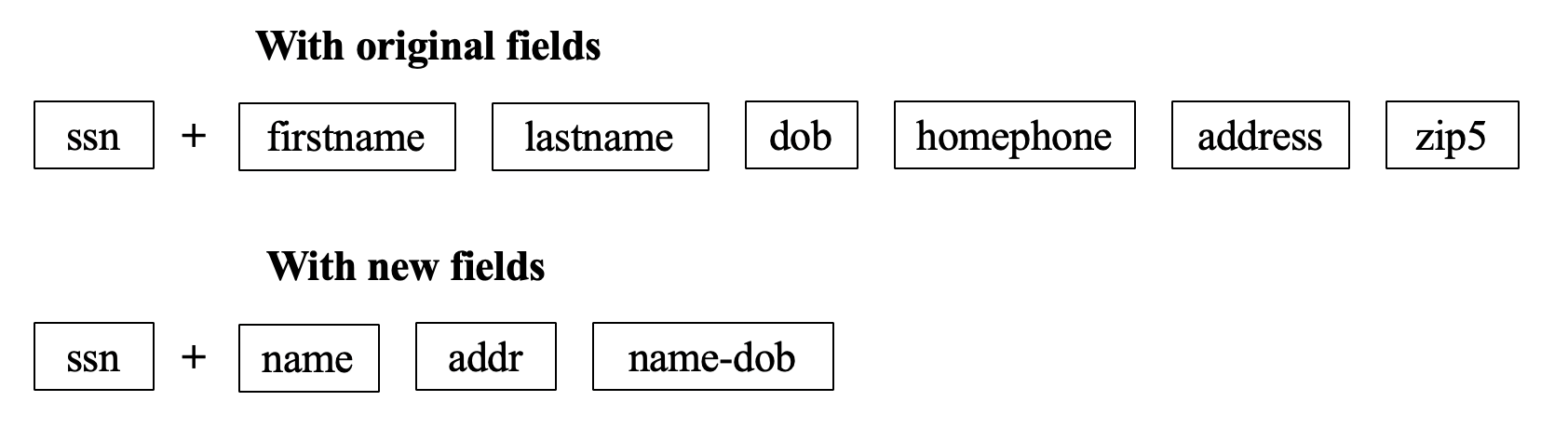
Some of the original fields are not unique identifiers of applicants. A large number of applicants share the same first name, last name or full name. Same street names and house numbers exist in different cities. Therefore, we want to create entities that contain more personally identifiable information. First, we create 3 new fields as follows:

* Concatenate “firstname” and “lastname” to create the **“name”** field.
* Concatenate “address” and “zip” to create the **“addr”** field.
* Concatenate “name” and “dob” to create the **“name-dob”** field.

Then with “name”, “dob”, “homephone” and “addr” fields, we create 10 different combinations so we can capture the different combinations of personal and contact information:



Then we combine “ssn”, which is the most personally identifiable field in the dataset, with the 6 original fields and 3 new fields:



At the end of this step, redundant fields are removed. In total, we have 6 entities and 19 combination groups:

* Entities: “name”, “ssn”, “dob”, “addr”, “homephone” and “name-dob”.
* Combinations: “name-homephone”, “name-addr”, “dob-homephone”, “dob-addr”, “homephone-addr”, “name-dob-homephone”, “name-dob-addr”, “name-homephone-addr”, “dob-homephone-addr”, “name-dob-homephone-addr”, “ssn-firstname”, “ssn-lastname”, “ssn-dob”, “ssn-homephone”, “ssn-address”, “ssn-zip5”, “ssn-name”, “ssn-addr” and “ssn-name-dob”.

##### Creating Candidate Variables

This section covers the logic and implementation details of creating the variables in Python, specifically how we make sure we only look back in time to calculate the value for each variable. Loop is used to automatically repeat the execution of variable creation within a predetermined range of fields and timeframes. First, we create 2 duplicate data frames (df1, df2), which contain “record”, “date” and all entities and combination groups. A list of “lags” in days is also generated, which includes 0, 1, 3, 7, 14 and 30 days.

* + 1. Days Since Variables  
       The first type of variable to create is days since variables, namely the number of days elapsed since the entity or combination group is last seen before the time of a particular application in a specific time point in the dataset. A small number often represents a high probability of fraud.

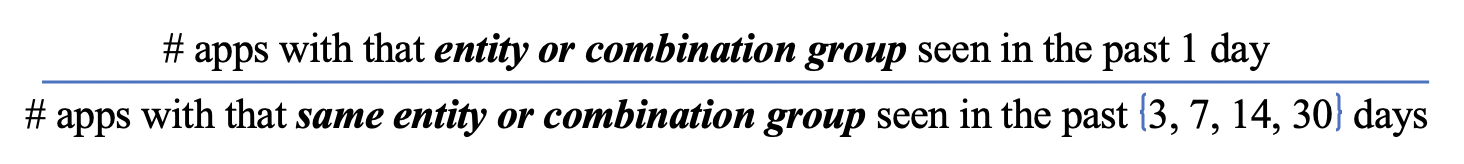
1. Create date\_lags Fields  
   For each number in the “lags” list, we concatenate it with “date” and generate a list of date\_lags, such as “date\_7”. Then we employ the timedelta function from the datetime package, add each lag number to each date.
2. Calculate the Number of Days  
   With each entity and combination group as a key, we perform an outer merge of df1 and df2 and generate df3. For such fields as “record” and “date” that appear in both df1 and df2, df3 treats fields from df1 as “record\_x”, “date\_x” and fields from df2 as “record\_y” and “date\_y”.  
     
   To ensure that only records before the time of a particular application are used, we set up a filter: “record\_x” > “record\_y”. Then the result is grouped by “record\_x”. The last date the entity or combination group is seen in the past time flow is extracted and used to calculate the number of days lapsed.
3. Replace Empty Values  
   However, the majority of records are not fraudulent. There are many empty values in this variable“#d\_since”, meaning that there is no same entity or combination group before a given time point. Therefore, we replace the empty values with the number of days elapsed since the first day of year 2016.   
     
   As a small number of days since an entity or combination is last seen represents a high occurrence of fraud, a larger number we replace the empty value with demonstrate low possibility of fraud.
   * 1. Velocity Variables

The second type of variable is velocity variables, which refers to the number of applications with the same entity of combination group within a specific time period. If the same entity or combination group is frequently seen within a given duration, it denotes potential fraudulent application.

We count the number of applications before the time of a particular application but within a specific timeframe. For these variables, a similar implementation method is used as the Days Since variable, but we not only filter records by record number, but also date.

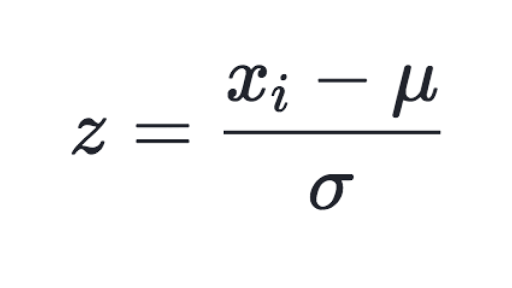
* + 1. Relative Velocity Variables

The third type of candidate variables is relative velocity variables, which is the ratio of short-term average number of applications versus long-term average number of applications. We calculate this variable according to the following formula:



##### Data Normalization

After variable creation, we Z-scale all variables to bring values to the same origin, which can further distinguish anomaly from normality. The formula of z-scaling is displayed below:



##### 4.4 Splitting Data

After normalizing all variables, we first remove the first 2 weeks of records from the dataset since the Days Since variables have limited days to look back into, so normal records are likely to stand out as fraud with a small value in this field during this period of time. And then we separate the last 2 months’ data as the out-of-time validation set, and split the remaining data into 80% training and 20% testing sets. We only perform feature selection with the training set.

#### 

#### Feature Selection

With the 286 variables created, we conduct feature selection with a two-step method. We first use univariate filters to select the most promising 50 variables, and then use a backward selection wrapper, Recursive Feature Elimination and Cross-Validation Selection, to reduce the dimensionality and keep the best number of variables.

##### Univariate Filters

We use two measures of goodness as filters: “Kolmogorov-Smirnov” (KS) score and Fraud Detection Rate (FDR). The KS score is used to measure how separate is one distribution from the other distribution. By calculating KS, we can measure how well a particular variable can distinguish the distribution of the normal records and the fraud records, or the label 0 and 1. The higher the KS score, the better performance of the variable in separating fraudulent records from normal records. The FDR score is the percentage of correctly detected frauds among records at a certain percentile. The higher the FDR score, the better performance of the variable.

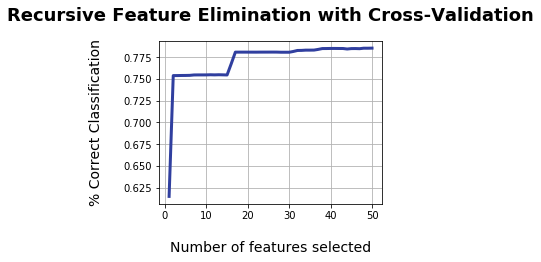
Filter methods are implemented in the following steps:

1. Create a new variable “random”, which is a sequence of random numbers generated, and we add a “random” variable as well as a “fraud\_label” variable along with other 286 variables to check if the KS and FDR measurements work efficiently. Ideally, “fraud\_label” should get perfect scores and “random” should get the worse.
2. Calculate the KS scores and FDR at 3% for all 288 variables and sort variables descendingly by the scores. Variable “fraud\_label” ranks at the top for both KS and FDR, and variable “random” ranks last for FDR, but ranks #275 for KS scores.
3. After removing the “fraud\_label” and “random” variables, we then take the average rank order of both to select the top 143 variables to process wrapper selection.

##### Wrapper

We use Recursive Feature Elimination and Cross-Validation Selection (RFECV) to reduce dimensionality based on cross validation scores.

A Logistic Regression model is used as a classifier in RFECV, and we run the model 3 times to drop 143 features stepwise to 50. Then we conducted our last run with 50 variables, and plot the number of features versus cross validation scores, see the picture below.

Though the optimal numbers selected by RFECV is 50, there is not much improvement in the cross validation scores of the model after around 20 features. Therefore, we pick the top 20 variables from RFECV ranking. The top 20 variables are: “addr-homephone\_#days\_since”, “addr-homephone\_lag14\_count”, “addr-homephone\_lag30\_count”, “addr-homephone\_lag7\_count”, “addr\_#days\_since”, “addr\_lag14\_count”, “addr\_lag1\_count”, “addr\_lag1\_lag14\_avg”, “addr\_lag30\_count”, “addr\_lag3\_count”, “addr\_lag7\_count”, “address\_#days\_since”, “address\_lag14\_count”, “address\_lag1\_count”, “address\_lag1\_lag14\_avg”, “address\_lag30\_count”, “address\_lag3\_count”, “address\_lag7\_count”, “homephone\_lag7\_count”, “name-dob\_#days\_since”

#### Model Algorithms

Four candidate models are chosen for this project, namely Logistic Regression, Boosted Tree, Random Forest, and Neural Net. We ultimately select one model based on classification performance. FDR at 3% is selected as the model measure of goodness in line with industry standard.

The general procedure of model training is as follows: First we fit a model with specific parameters on the training set and then use the model to predict the fraud probability of each record as a fraud score on all the training, testing and out-of-time sets. Next, for each set, we sort all records by the predicted fraud scores, and take the top 3% with the highest fraud score and count how many of these records are truly fraud applications, and calculate the percentage of these accurately detected fraud records among all fraud records in each set as the FDR rate at 3% for that data set. We then change the parameters of the model and repeat the procedure to find the best performing version of the model. Each version of each model is run 5 times, so that we can take an average of the FDR to account for the randomness factor in model training.

##### Logistic Regression

Logistic Regression uses maximum likelihood method to fit the logistic function, and gives outputs of the probability of fraud. Mathematically, a Logistic Regression model gives a linear fit to log-odds, and estimates the coefficients to yield a number close to 1 for each possible fraud application, and a number close to 0 for normal applications.

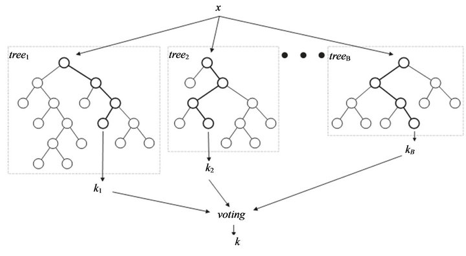
##### Boosted Tree

A Boosted Tree is a linear combination of weak learners or models to result in a strong learner. Each weak learner is trained to predict the residual error of the current sum. Specifically, the algorithm firstly builds a weak model to predict the output, and this first model is the closest approximation of the output. Based on the first model, the second weak model is built to predict the residual error of the first model. Then the third model is trained to predict the residual error of the previous two models summed up. The same logic applies to all the weak models. In the end, we get a strong model by summing up all the weak models.

For this project, we try out different numbers of trees or models. The “max depth” parameter, which reflects the complexity of each weak tree, is set to be smaller than 5 as we prefer more weak and simple trees rather than a few complex trees. We also set the loss function for regularization to perform embedded feature selection to account for the complexity of the Boosted Tree algorithm.

##### Random Forest

A Random Forest model is an ensemble model, which builds many complex independent decision trees. Each tree is a full model to predict the output and is a strong learner. The algorithm uses a different subset of variables when building each individual tree and/or during each split iteration. These randomness associated with the random forest algorithm makes all the trees relatively independent from each other. Voting and averaging are two of the common ways to combine all the strong trees, which cancel out the biases and therefore does not suffer from the overfitting problem. The Random Forest algorithm is usually robust, stable, and easily-generalizable.

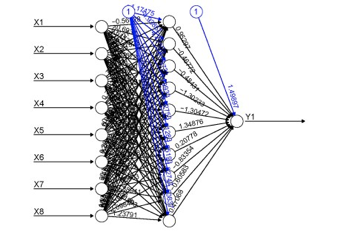


For this project, we try out different versions of the model with different numbers of trees or models and ‘minimum samples leaf’, which is the minimum samples required to be at a leaf node for a split point.

##### Neural Net

A Neural Net model consists of an input layer, at least one hidden layer and an output layer. The input layer has many nodes, each node receiving information from an specific independent variable respectively. The dependent variable Y is the output layer. Typically, there is only one scalar output Y. If there is a vector of Ys, multiple classifications will be used for multiple outputs. Each node in a hidden layer receives weighted signals from all the nodes in the previous layer and does a transformation on this linear combination of signals. Usually, we use logistic regression function as the transformation function.

Input Layer Hidden Layer Output Layer



The weights of signals are what is needed to be trained by back-propagating the error, record by record. Firstly, the Neural Net model initializes a random combination of weights. When the first record is pushed through the neural net algorithm, the model transforms the linear combination of signals with the current weights using a specified transformation function like a logistic regression, predicts the output and calculates the error. A typical error function is as follows:



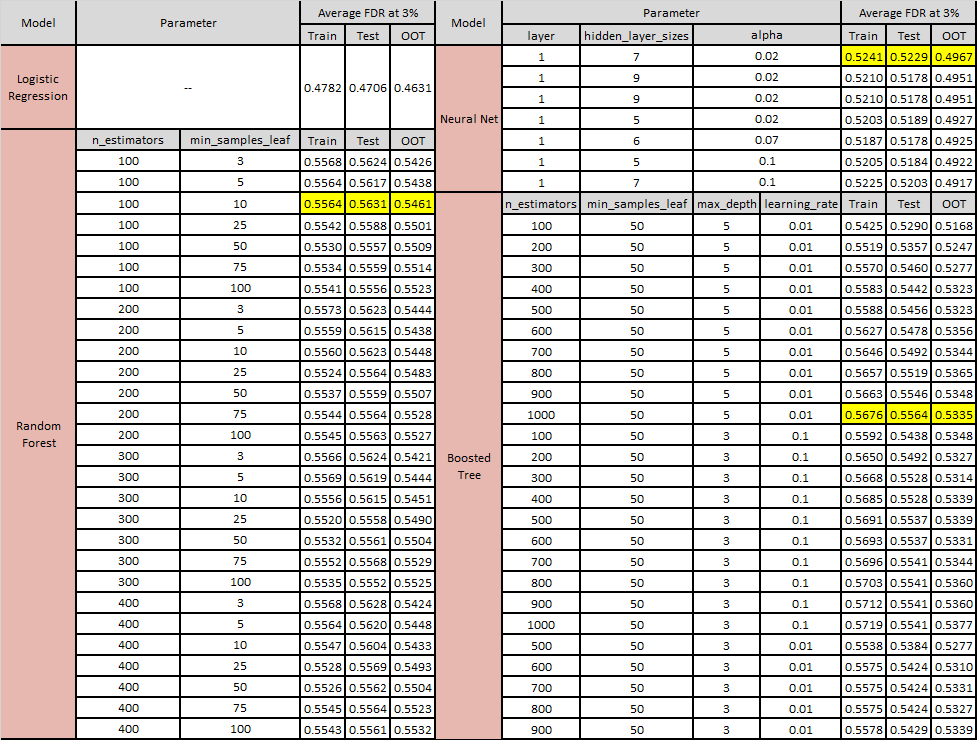
Then, according to the objective function (e.g to minimize error E(a)), the Neural Net algorithm will take the derivative to calculate the gradient of the error with respect to the node weights, and then propagate them back to slightly adjust weights. With each data record pushed through the algorithm one by one, all the weights will be gradually adjusted. The entire data set will be passed through the algorithm many times as the weights settle into a local optimum. How fast the weights are adjusted depends on the learning rate. The following formula shows how the weights are adjusted.



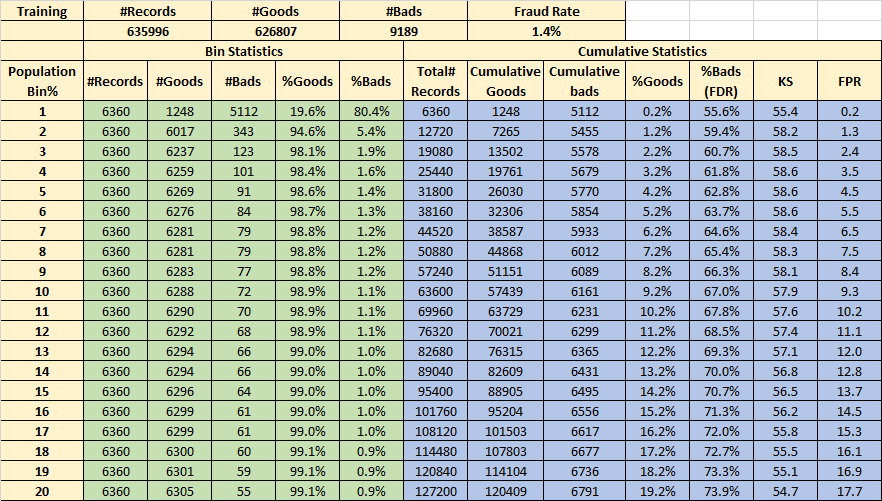
For this project, we use only one hidden layer and try out different numbers of nodes in this layer with different alpha levels or learning rates.

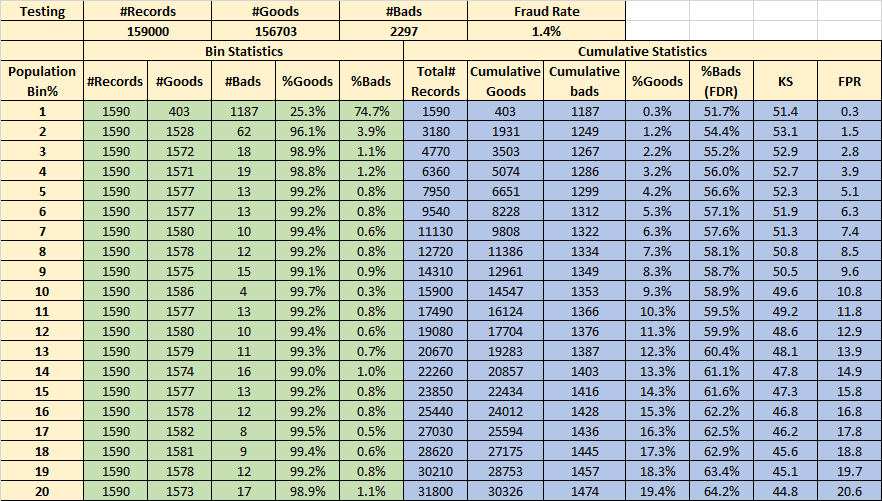
#### Results

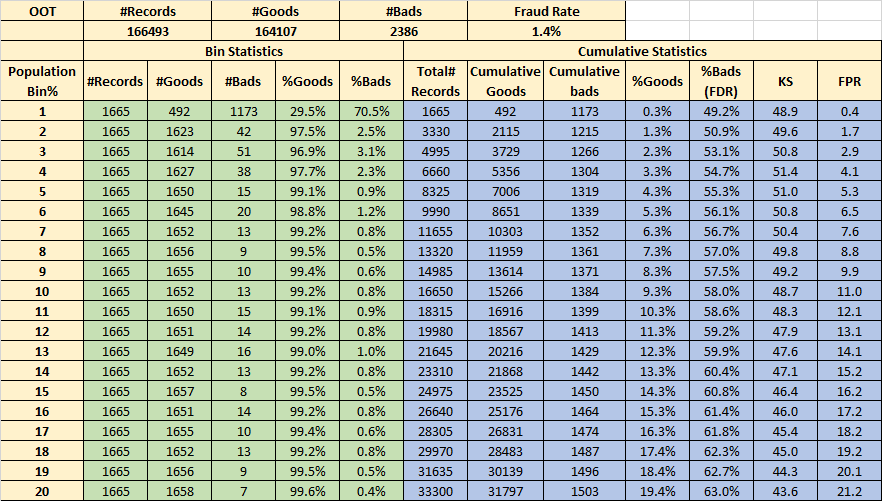
After building the above models and tuning the parameters, we compare the results of each model (shown below). We select the Random Forest model (n\_estimators = 100, min\_samples\_leaf=10) as the final model since it has the highest 3% FDR (56.3%) in the testing set.



We run the selected Random Forest model on the training, testing, and out-of-time sets again to score each record’s possibility of fraud (shown below). We split each data set into 100 bins and the tables below show the FDR among the highest-ranking 20% in terms of fraud possibility with bin statistics and the cumulative statistics.







#### 

#### Conclusion

For this project, we firstly explore and assess the data to understand the characteristics of the application data. From the data quality report we see common frivolous data entry, and we clean the data by replacing the frivolous values by the “record” value to make sure the commonality in frivolousness is not confused as fraud by our model.

Based on the characteristics of fraud behaviour, we build 286 candidate variables out of the 8 original fields (except record number and fraud label). Then, we scale all the candidate variables and do feature selection. We use KS score and FDR at 3% as the measure of goodness to filter out 143 candidate variables, and then use a wrapper to select the final 20 variables for model building.

We use the selected variables to build the baseline linear model (Logistic Regression) and three nonlinear models (Neural Net, Boosting Tree and Random Forest). After tuning the parameters and comparing the results of each model, we select Random Forest as our final model as it has the highest FDR at 3% for the testing set. In the end, we build FDR tables to show the performance of this final model.

If we were granted more time, we would like to improve in the following areas. First, we would consult with application fraud experts to better understand patterns of fraud behaviour, so that we can build more predicative candidate variables to improve model performance. Second, we would further fine tune the parameters of the four candidate models and try more nonlinear algorithms such as SVM for a wider selection for the final model.

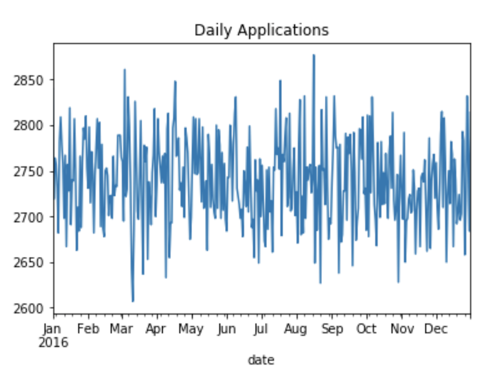
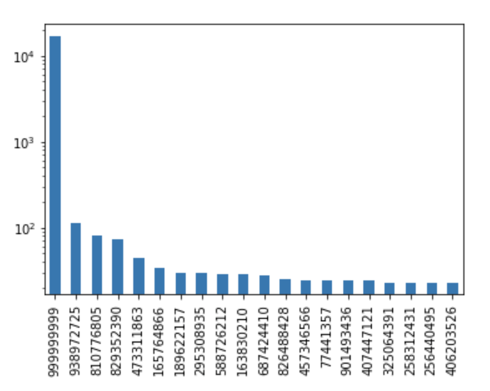
##### 

#### Appendix

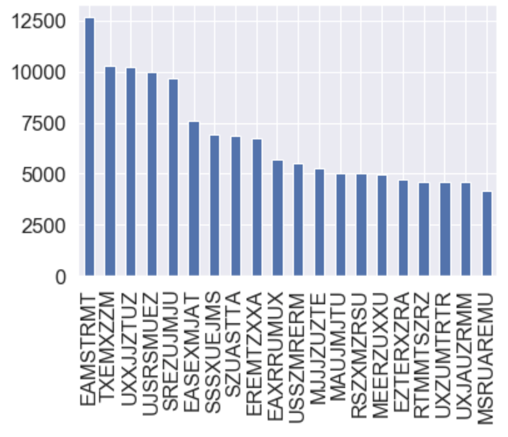
**Data Quality Report on Application Fraud Data**

1. **Data Description**The Application Fraud data is a dataset emulating the nature of real fraud data. The data file contains one million records of application information in the 2016, with 8 columns of personal information and one fraud label column indicating whether an application has turned out to be fraud or not.
2. **Summary Tables of Fields**

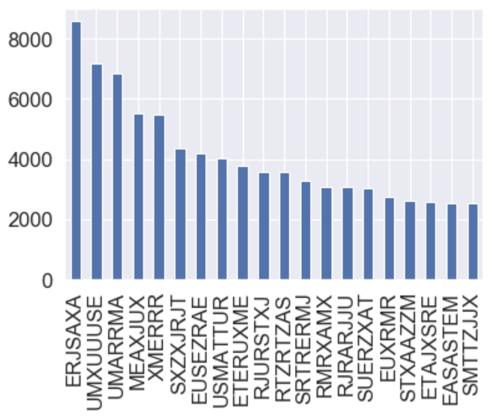
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Field No. | Field Name | Data Type | Field  Type | # of Populated Records | % of Populated Records | # of Unique Values | # of Zeros | Most Common Field Value | Number of Occurrence |
| 1 | record | int64 | Categorical | 1,000,000 | 100.00% | 1,000,000 | 0 | N/A | N/A |
| 2 | date | int64 | Categorical | 1,000,000 | 100.00% | 365 | 0 | 2016-01-01 | 19,276 |
| 3 | ssn | int64 | Categorical | 1,000,000 | 100.00% | 835,819 | 0 | 999999999 | 16,935 |
| 4 | firstname | object | Categorical | 1,000,000 | 100.00% | 78,136 | 0 | EAMSTRMT | 12,658 |
| 5 | lastname | object | Categorical | 1,000,000 | 100.00% | 177,001 | 0 | ERJSAXA | 8,580 |
| 6 | address | object | Categorical | 1,000,000 | 100.00% | 828,774 | 0 | 123 MAIN ST | 1,079 |
| 7 | zip5 | int64 | Categorical | 1,000,000 | 100.00% | 26,370 | 0 | 68138 | 823 |
| 8 | dob | int64 | Categorical | 1,000,000 | 100.00% | 42,673 | 0 | 19070626 | 126,568 |
| 9 | homephone | int64 | Categorical | 1,000,000 | 100.00% | 28,244 | 0 | 9999999999 | 78,512 |
| 10 | fraud\_label | int64 | Categorical | 1,000,000 | 100.00% | 2 | 985,607 | 0 | 985,607 |

1. **Field Descriptions**
   1. date  
      Dates of the applications range from 2016-01-01 to 2016-12-31. Since the number of occurrences of each date is the same as the number of applications of each day, we plot the number of daily applications for an overview of the distribution of dates and applications. Since 2016 is not an ecliptic year, we fill the date 2016-02-29 with the same number of applications on 2016-02-28 to remove the dip in the data.
   2. ssn  
      The social security number filed by applicants. The distribution of the top 15 SSNs with the most occurrences is as below. 999999999 has the most occurrences, which indicates that a lot of people filled in this field frivolously.
   3. firstname

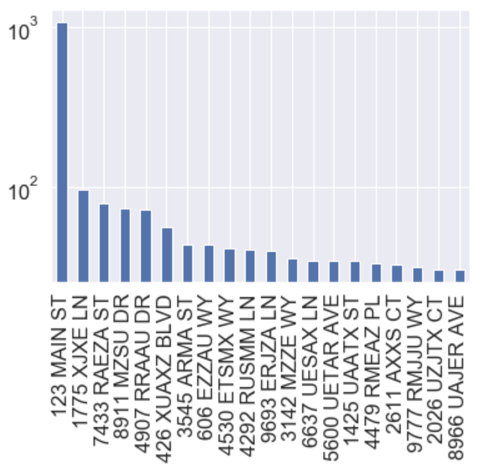
The first name of the applicant. The distribution of the top 20 values with the most occurrences is shown below.



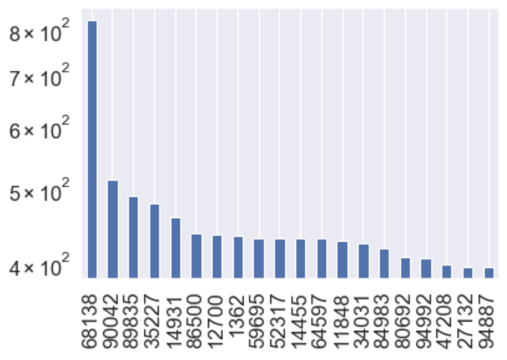
* 1. lastname  
     The last name of the applicant. The distribution of the top 20 values with the most occurrences is shown below.



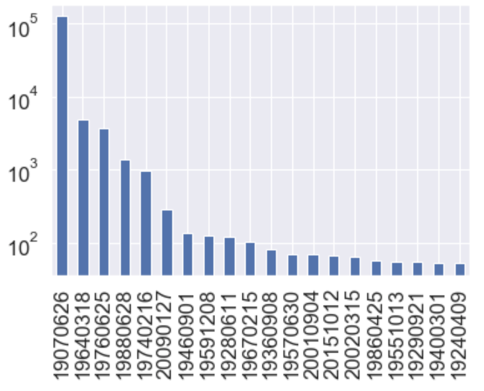
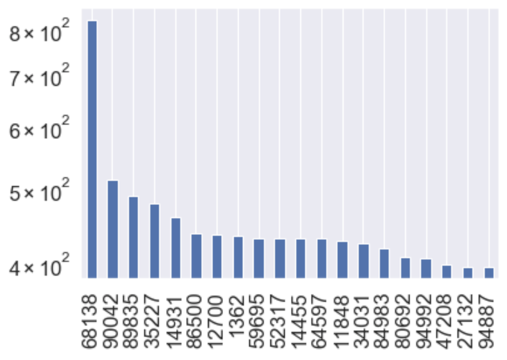
* 1. address  
     The addresses given by the applicants for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 123 MAIN ST is the most common address and has ten times more occurrences than the second ranking address. It is very likely to be a result of applicants filling in addresses frivolously.



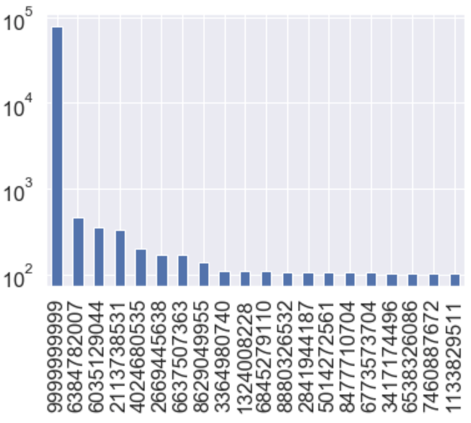
* 1. zip5  
     The 5-digit zip code given by the applicant for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 68138 is the most common zip code and has significantly more occurrences than the second ranking zip code. It is likely to be a result of applicants filling in information frivolously, or an indication of possible fraud applications.



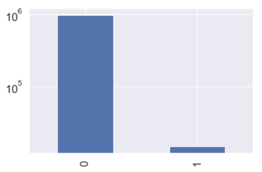
* 1. dob  
     Date of birth given for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 19070626 is the most common date of birth. It is likely to be a result of applicants filling in information frivolously, or an indication of possible fraud applications.



* 1. homephone  
     Landline at home given for the application purpose. The distribution of the top 20 values with the most occurrences is shown below. 9999999999 is the most common phone number. It is likely to be a result of applicants filling in information frivolously.



* 1. fraud\_label  
     The indicator variable of whether the record later turned out to be a fraud application. Zeros indicate normal applications, and ones indicate fraud applications. The majority of the records were non-fraud applications, which accounts for 98.56% of the entire dataset. There are 14,393 fraud records in total, which account for about 1.44%.



**Candidate Variables for the First 5 Records**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **record** | 1 | 2 | 3 | 4 | 5 |
| **1** | **ssn\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **2** | **ssn\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **3** | **ssn\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **4** | **ssn\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **5** | **ssn\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **6** | **ssn\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **7** | **ssn\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **8** | **ssn\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **9** | **ssn\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **10** | **ssn\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **11** | **ssn\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **12** | **address\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **13** | **address\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **14** | **address\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **15** | **address\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **16** | **address\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **17** | **address\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **18** | **address\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **19** | **address\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **20** | **address\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **21** | **address\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **22** | **address\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **23** | **dob\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **24** | **dob\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **25** | **dob\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **26** | **dob\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **27** | **dob\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **28** | **dob\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **29** | **dob\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **30** | **dob\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **31** | **dob\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **32** | **dob\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **33** | **dob\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **34** | **homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **35** | **homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **36** | **homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **37** | **homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **38** | **homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **39** | **homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **40** | **homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **41** | **homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **42** | **homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **43** | **homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **44** | **homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **45** | **name\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **46** | **name\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **47** | **name\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **48** | **name\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **49** | **name\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **50** | **name\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **51** | **name\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **52** | **name\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **53** | **name\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **54** | **name\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **55** | **name\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **56** | **addr\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **57** | **addr\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **58** | **addr\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **59** | **addr\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **60** | **addr\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **61** | **addr\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **62** | **addr\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **63** | **addr\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **64** | **addr\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **65** | **addr\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **66** | **addr\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **67** | **name-dob\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **68** | **name-dob\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **69** | **name-dob\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **70** | **name-dob\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **71** | **name-dob\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **72** | **name-dob\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **73** | **name-dob\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **74** | **name-dob\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **75** | **name-dob\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **76** | **name-dob\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **77** | **name-dob\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **78** | **name-addr\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **79** | **name-addr\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **80** | **name-addr\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **81** | **name-addr\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **82** | **name-addr\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **83** | **name-addr\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **84** | **name-addr\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **85** | **name-addr\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **86** | **name-addr\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **87** | **name-addr\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **88** | **name-addr\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **89** | **name-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **90** | **name-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **91** | **name-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **92** | **name-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **93** | **name-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **94** | **name-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **95** | **name-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **96** | **name-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **97** | **name-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **98** | **name-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **99** | **name-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **100** | **dob-addr\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **101** | **dob-addr\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **102** | **dob-addr\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **103** | **dob-addr\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **104** | **dob-addr\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **105** | **dob-addr\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **106** | **dob-addr\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **107** | **dob-addr\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **108** | **dob-addr\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **109** | **dob-addr\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **110** | **dob-addr\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **111** | **dob-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **112** | **dob-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **113** | **dob-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **114** | **dob-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **115** | **dob-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **116** | **dob-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **117** | **dob-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **118** | **dob-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **119** | **dob-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **120** | **dob-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **121** | **dob-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **122** | **addr-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **123** | **addr-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **124** | **addr-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **125** | **addr-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **126** | **addr-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **127** | **addr-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **128** | **addr-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **129** | **addr-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **130** | **addr-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **131** | **addr-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **132** | **addr-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **133** | **name-dob-addr\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **134** | **name-dob-addr\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **135** | **name-dob-addr\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **136** | **name-dob-addr\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **137** | **name-dob-addr\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **138** | **name-dob-addr\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **139** | **name-dob-addr\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **140** | **name-dob-addr\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **141** | **name-dob-addr\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **142** | **name-dob-addr\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **143** | **name-dob-addr\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **144** | **name-dob-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **145** | **name-dob-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **146** | **name-dob-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **147** | **name-dob-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **148** | **name-dob-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **149** | **name-dob-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **150** | **name-dob-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **151** | **name-dob-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **152** | **name-dob-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **153** | **name-dob-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **154** | **name-dob-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **155** | **name-addr-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **156** | **name-addr-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **157** | **name-addr-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **158** | **name-addr-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **159** | **name-addr-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **160** | **name-addr-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **161** | **name-addr-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **162** | **name-addr-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **163** | **name-addr-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **164** | **name-addr-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **165** | **name-addr-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **166** | **dob-addr-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **167** | **dob-addr-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **168** | **dob-addr-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **169** | **dob-addr-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **170** | **dob-addr-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **171** | **dob-addr-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **172** | **dob-addr-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **173** | **dob-addr-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **174** | **dob-addr-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **175** | **dob-addr-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **176** | **dob-addr-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **177** | **name-dob-addr-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **178** | **name-dob-addr-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **179** | **name-dob-addr-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **180** | **name-dob-addr-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **181** | **name-dob-addr-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **182** | **name-dob-addr-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **183** | **name-dob-addr-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **184** | **name-dob-addr-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **185** | **name-dob-addr-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **186** | **name-dob-addr-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **187** | **name-dob-addr-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **188** | **ssn-firstname\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **189** | **ssn-firstname\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **190** | **ssn-firstname\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **191** | **ssn-firstname\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **192** | **ssn-firstname\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **193** | **ssn-firstname\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **194** | **ssn-firstname\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **195** | **ssn-firstname\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **196** | **ssn-firstname\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **197** | **ssn-firstname\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **198** | **ssn-firstname\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **199** | **ssn-lastname\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **200** | **ssn-lastname\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **201** | **ssn-lastname\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **202** | **ssn-lastname\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **203** | **ssn-lastname\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **204** | **ssn-lastname\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **205** | **ssn-lastname\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **206** | **ssn-lastname\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **207** | **ssn-lastname\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **208** | **ssn-lastname\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **209** | **ssn-lastname\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **210** | **ssn-address\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **211** | **ssn-address\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **212** | **ssn-address\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **213** | **ssn-address\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **214** | **ssn-address\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **215** | **ssn-address\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **216** | **ssn-address\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **217** | **ssn-address\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **218** | **ssn-address\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **219** | **ssn-address\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **220** | **ssn-address\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **221** | **ssn-zip5\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **222** | **ssn-zip5\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **223** | **ssn-zip5\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **224** | **ssn-zip5\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **225** | **ssn-zip5\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **226** | **ssn-zip5\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **227** | **ssn-zip5\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **228** | **ssn-zip5\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **229** | **ssn-zip5\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **230** | **ssn-zip5\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **231** | **ssn-zip5\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **232** | **ssn-dob\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **233** | **ssn-dob\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **234** | **ssn-dob\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **235** | **ssn-dob\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **236** | **ssn-dob\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **237** | **ssn-dob\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **238** | **ssn-dob\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **239** | **ssn-dob\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **240** | **ssn-dob\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **241** | **ssn-dob\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **242** | **ssn-dob\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **243** | **ssn-homephone\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **244** | **ssn-homephone\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **245** | **ssn-homephone\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **246** | **ssn-homephone\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **247** | **ssn-homephone\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **248** | **ssn-homephone\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **249** | **ssn-homephone\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **250** | **ssn-homephone\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **251** | **ssn-homephone\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **252** | **ssn-homephone\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **253** | **ssn-homephone\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **254** | **ssn-name\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **255** | **ssn-name\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **256** | **ssn-name\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **257** | **ssn-name\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **258** | **ssn-name\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **259** | **ssn-name\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **260** | **ssn-name\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **261** | **ssn-name\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **262** | **ssn-name\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **263** | **ssn-name\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **264** | **ssn-name\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **265** | **ssn-addr\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **266** | **ssn-addr\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **267** | **ssn-addr\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **268** | **ssn-addr\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **269** | **ssn-addr\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **270** | **ssn-addr\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **271** | **ssn-addr\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **272** | **ssn-addr\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **273** | **ssn-addr\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **274** | **ssn-addr\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **275** | **ssn-addr\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |
| **276** | **ssn-name-dob\_#d\_since** | 0 | 0 | 0 | 0 | 0 |
| **277** | **ssn-name-dob\_ps\_0d\_count** | 1 | 1 | 1 | 1 | 1 |
| **278** | **ssn-name-dob\_ps\_1d\_count** | 1 | 1 | 1 | 1 | 1 |
| **279** | **ssn-name-dob\_ps\_3d\_count** | 1 | 1 | 1 | 1 | 1 |
| **280** | **ssn-name-dob\_ps\_7d\_count** | 1 | 1 | 1 | 1 | 1 |
| **281** | **ssn-name-dob\_ps\_14d\_count** | 1 | 1 | 1 | 1 | 1 |
| **282** | **ssn-name-dob\_ps\_30d\_count** | 1 | 1 | 1 | 1 | 1 |
| **283** | **ssn-name-dob\_ps\_1d\_ps3d\_avg** | 3 | 3 | 3 | 3 | 3 |
| **284** | **ssn-name-dob\_ps\_1d\_ps7d\_avg** | 7 | 7 | 7 | 7 | 7 |
| **285** | **ssn-name-dob\_ps\_1d\_ps14d\_avg** | 14 | 14 | 14 | 14 | 14 |
| **286** | **ssn-name-dob\_ps\_1d\_ps30d\_avg** | 30 | 30 | 30 | 30 | 30 |